Intro to Distributed Systems

H. Levy

Distributed Systems

- **Nearly all systems today are distributed in some way, e.g:**
  - the use email
  - they access files over a network
  - they access printers over a network
  - they are backed up over network
  - they share other physical or logical resources
  - they cooperate with other people on other machines
  - soon: they receive video, audio, etc.

Why use distributed systems?

- **Distributed systems are now a requirement:**
  - economics dictate that we buy small computers
  - everyone needs to communicate
  - we need to share physical devices (printers) as well as information (files, etc.)
  - many applications are by their nature distributed (bank teller machines, airlines reservations, ticket purchasing)
  - in the future, to solve the largest problems, we will need to get large collections of small machines to cooperate together (parallel programming)

What is a distributed system?

- **There are several levels of distribution.**
- **Earliest systems used simple explict network pograms:**
  - FTP : file transfer program
  - Telnet (rlogin) : remote login program
  - mail
  - remote job entry (or rsh) : run jobs remotely
- **Each system was a completely autonomous independent system, connected to others on the network.**

Loosely-Coupled Systems

- **Most distributed systems are "loosely-coupled".**
- **Each CPU runs an independent autonomous OS.**
- **Hosts communicate through message passing.**
- **Computers don't really trust each other.**
- **Some resources are shared, but most are not.**
- **The system may look differently from different hosts.**
- **Typically, communication times are long.**

Closely-Coupled Systems

- **A distributed system becomes more "closely coupled" as it:**
  - appears more uniform in nature
  - runs a "single" operating system
  - has a single security domain
  - shares all logical resources (e.g., files)
  - shares all physical resources (CPUs, memory, disks, printers, etc)
- **In the limit, a distributed system looks to the user as if it were a centralized timesharing system, except that it's constructed out of a distributed collection of hardware and software components.**

Tightly-Coupled Systems

- A "tightly-coupled" system usually refers to a multiprocessor.
  - runs a single copy of the OS with a single job queue
  - has a single address space
  - usually has a single bus or backplane to which all processors and memories are connected
  - has very low communication latency
  - processors communicate through shared memory

7

Some Issues in Distributed Systems

- Transparency (how visible is the distribution)
- Security
- Reliability
- Performance
- Scalability
- Programming models
- Communications models

8

Transparency

- In a true distributed system with transparency:
  - it would appear as a single system
  - different nodes would be invisible
  - jobs would migrate automatically from node to node
  - a job on one node would be able to use memory on another

9

Distribution and the OS

- There are various issues that the OS must deal with:
  - how to provide efficient network communication
  - what protocols to use
  - what is the application interface to remote apps (although this might be a language issue)
  - protection of distributed resources

10

The Network

- There are various network technologies that can be used to interconnect nodes.
- In general, Local Area Networks (LANs) are used to connect hosts within a building. Wide Area Networks (WANs) are used across the country or planet.
- We are at an interesting point, as network technology is about to see an order-of-magnitude performance increase. This will have a huge impact on the kinds of systems we can build.

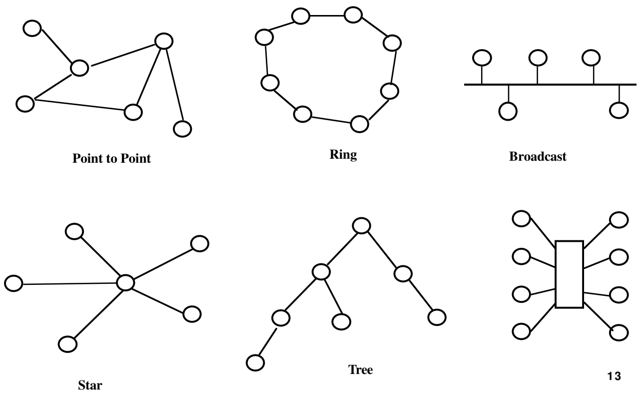
11

Issues in Networking

- Routing
- Bandwidth and contention
- Latency
- Reliability
- Efficiency
- Cost
- Scalability

12

Network Topologies



Two ways to handle networking

- **Circuit Switching**
  - what you get when you make a phone call
  - good when you require constant bit rate
  - good for reserving bandwidth (refuse connection if bandwidth not available)
- **Packet Switching**
  - what you get when you send a bunch of letters
  - network bandwidth consumed only when sending
  - packets are routed independently
  - packetizing may reduce delays (using parallelism)

Packet switching is preferable for data communications

- **From the perspective of the network**
  - but may not be preferable for some applications
- **Applications are bursty**
  - variable amounts of info at irregular intervals
  - a diskless workstation: needs all bandwidth to transfer a page, so can't reserve it
  - circuit switching may have high cost to set up connection
  - mantaining the connection may waste bandwidth if connection is used infrequently

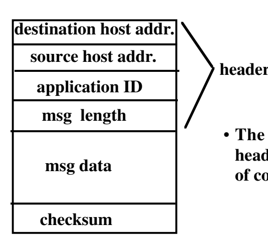
New Applications

- **Video and Voice may be different (more like phone system)**
- **But with data compression, makes circuit switching less attractive:**
  - compressed video generates a variable bit rate signal
  - signal needs to be transported within a certain max. delay, but bandwidth needed is variable
- **New applications will be very bursty and will require guarantees about latency.**

Messages

- **At a low level, network communication is via *messages*.**
- **A message is simply a typed byte string passed between two levels of the system (e.g., OS to OS, app to app).**
- **A message usually contains a header, indicating what kind of information it contains, and some data.**
- **What the message "means," i.e., how to interpret the bytes in the message, is an *agreement* between the two communicating parties (the protocol).**

Messages

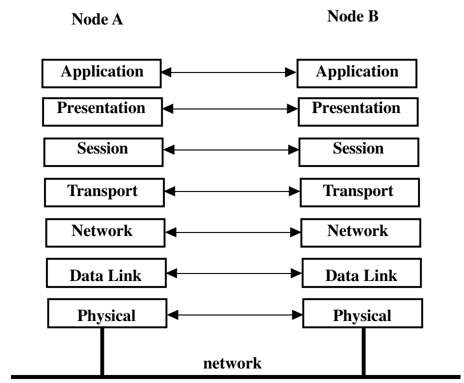


- **The msg data may itself contain a header and some data for another level of communication, and so on.**

The OSI Model

- The Open Systems Interconnect model is a standard way of understanding the conceptual layers of network communication.
- This is a model, nobody builds systems like this.
- Each level provides certain functions and guarantees, and communicates with the *same* level on remote nodes.
- A message is generated at the highest level, and is passed down the levels, encapsulated by lower levels, until it is sent over the wire.
- On the destination, it makes its way up the layers, until the high-level msg reaches its high-level destination.

OSI Levels



OSI Levels

- Physical Layer: electrical details of bits on the wire
- Data Link: sending “fames” of bits and error detection
- Network Layer: routing packets to the destination
- Transport Layer: reliable transmission of messages, disassembly/assembly, ordering, retransmission of lost packets
- Session Layer: really part of transport, typ. not impl.
- Presentation Layer: data representation in the message
- Application: high-level protocols (mail, ftp, etc)